Processing In Memory

New Models for Future Architectures

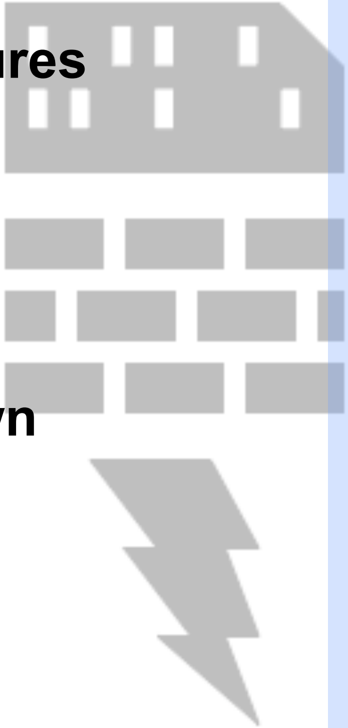
December 12, 2006

Arun Rodrigues
Org. 1423

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy's National Nuclear Security Administration
under contract DE-AC04-94AL85000.

Dusty Decks, Memory Walls & The Speed of Light

- SW resists change
  - Limit new Architectures
  - Legacy HW can't support new SW models
  - "Chicken & Egg"
- "Memory Wall" is known problem
- Speed of Light – fundamental problem



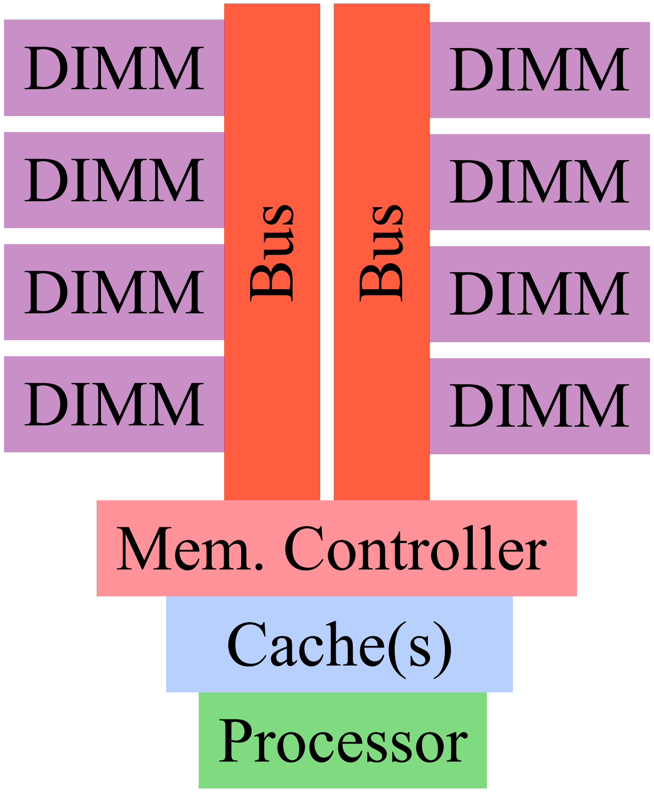
Outline

- Conventional Memory
- Bottlenecks & Impacts
- Processing-in-Memory
  - Chip Designs
  - Fabrication
  - System Design
- Programming

Conventional Memory Hierarchy

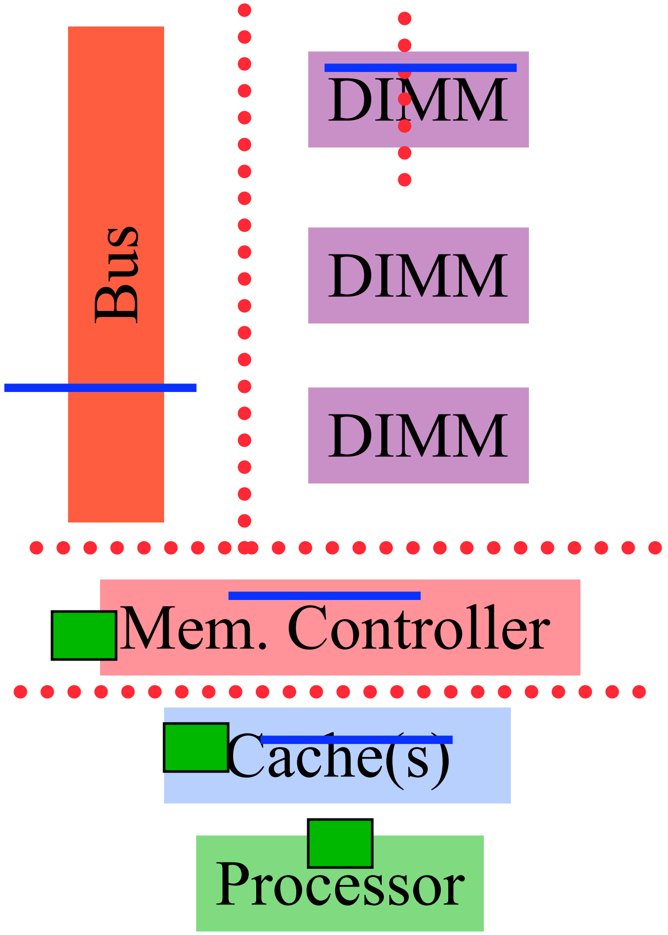
- Processors connected to the caches…
- Caches connected to the MC…
- MC's connected to the DIMM Bus…

- Processor & Cache (usually) on same chip
- MC sometimes on-chip
- MC converts from address to DRAM commands, reorders requests to maximize locality, arbitrate channels/busses



Bottlenecks & Complexity

- Chip/Board boundaries
- Bus/Bank/Row contention
- Coherency
- Complexity
  - OOO Memory Queues
  - MC command reordering
  - Prefetching collisions
  - Cache non-determinism
- Results in high latency
  - O(100) ns



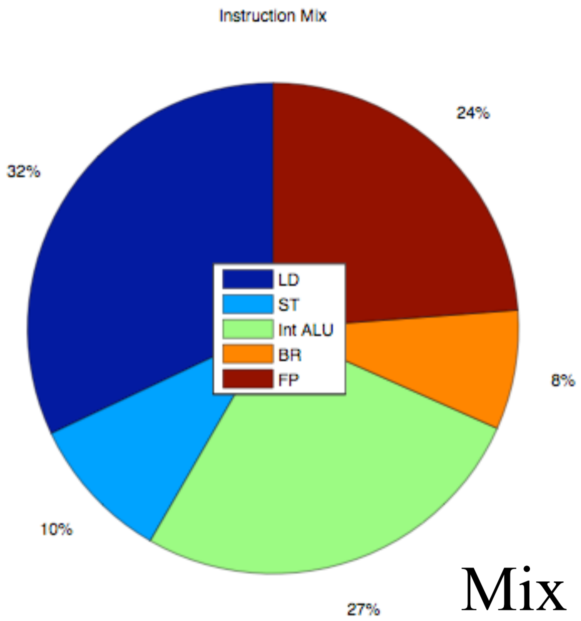
Sandia National Laboratories

Latency Impact

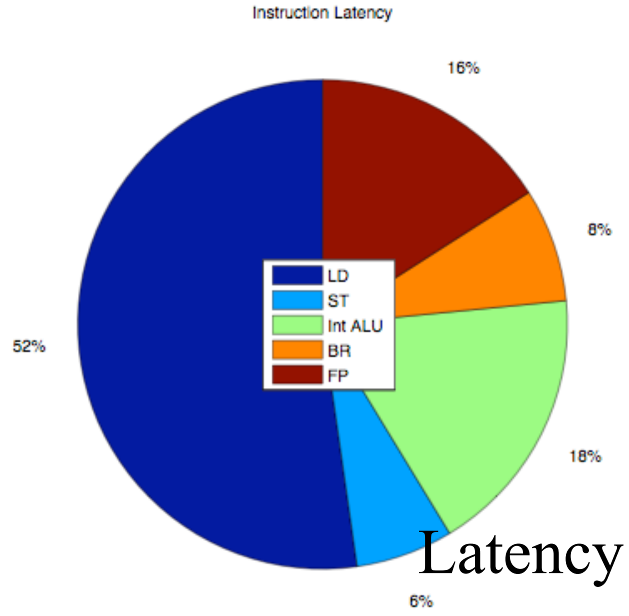
- **Loads often largest category of instruction**
- **Tend to dominate latency**
- **Solutions:**
  - **Caches**
  - **Prefetch**
  - **MC reordering**
  - **OOE**
  - **SW prefetch**
- **Result?**

Load	43%
Store	14%
Integer	27%
Branch	9%
FP	7%

Sandia Mix



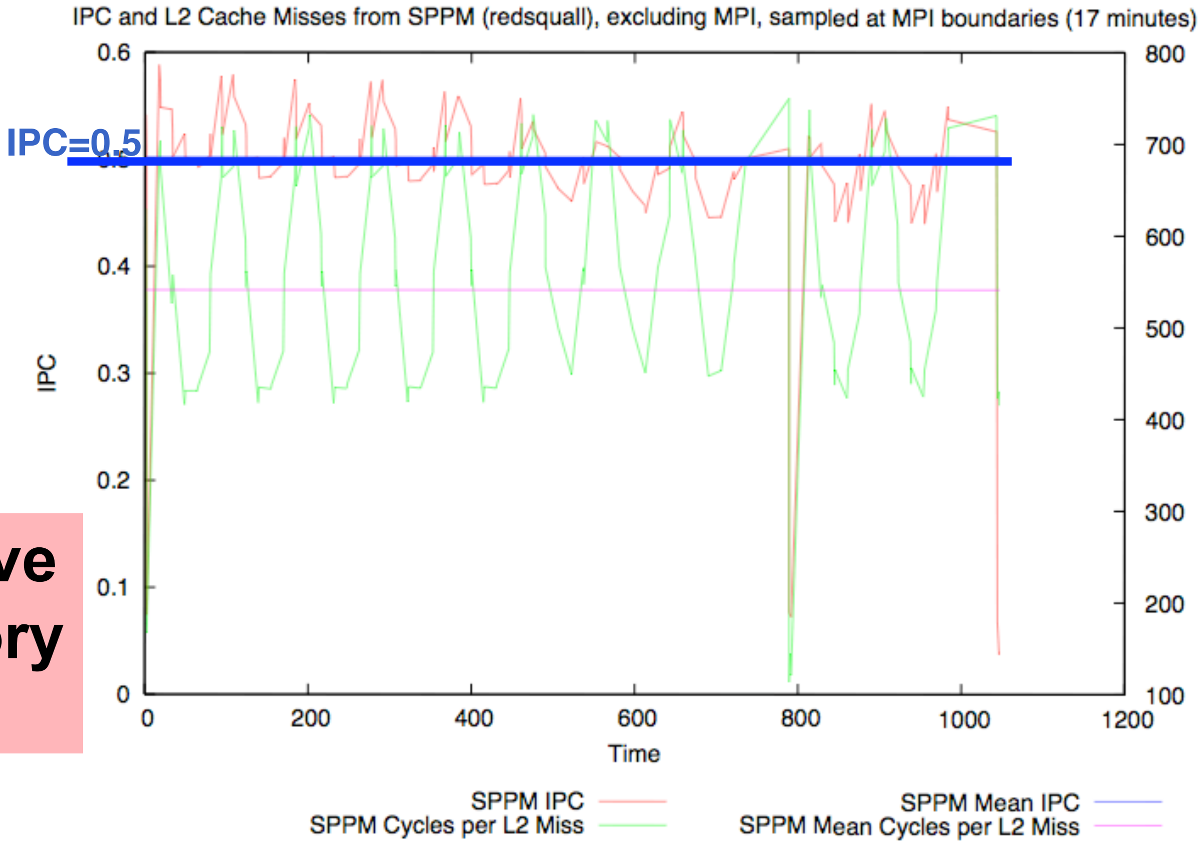
Mix



Latency

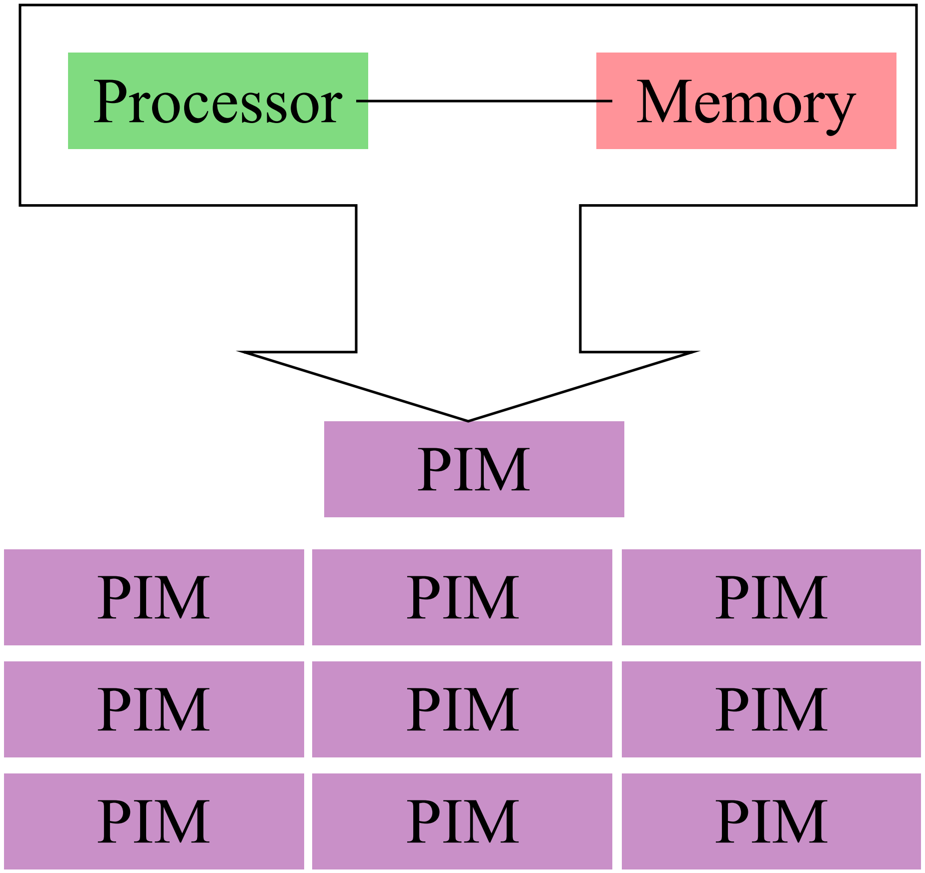
Latency Impact

- Low IPC
  - CTH: 0.37
  - sPPM: 0.5
  - LAMMPs: 0.5
- Strong correlation between cache misses & low IPC
- Even with aggressive HW features, memory dominates



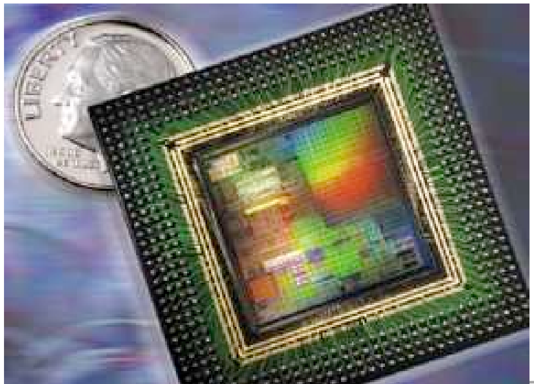
PIM: The Solution

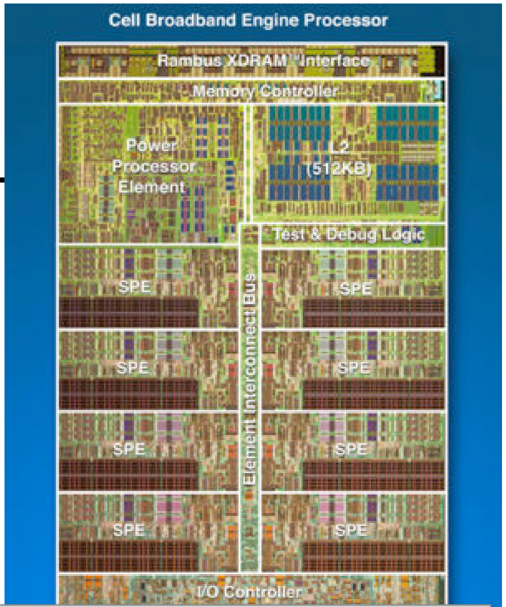
- If there is a bottleneck, go around it!
- Combine processor and memory
- Processors (logic) cheap, latency is expensive
- "Single Part" computer
- Simplify
  - No/Small caches
  - In-order
  - Massive Parallelism

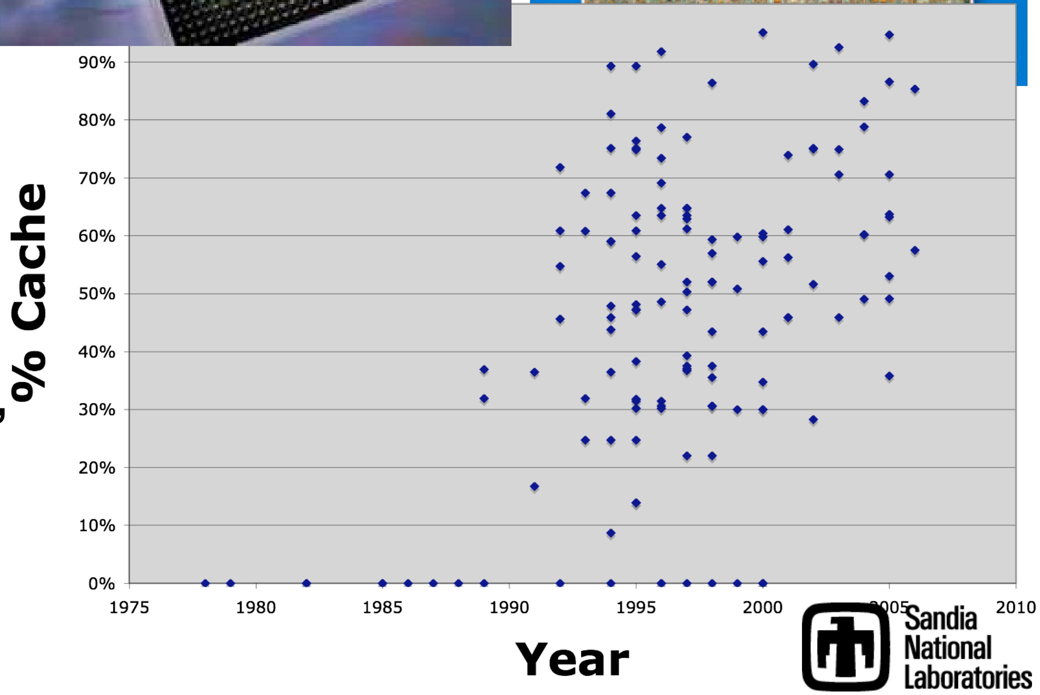


PIMs Conquer All

- Embedded Systems
  - eDRAM IP blocks
- Game systems
  - CELL, Wii, GameCube, and Xbox 360, PS2
- Conventional Processors
  - Caches often >80% transistor count
- PIM Projects
  - Execube, DIVA, PIMLite, HTMT, IRAM

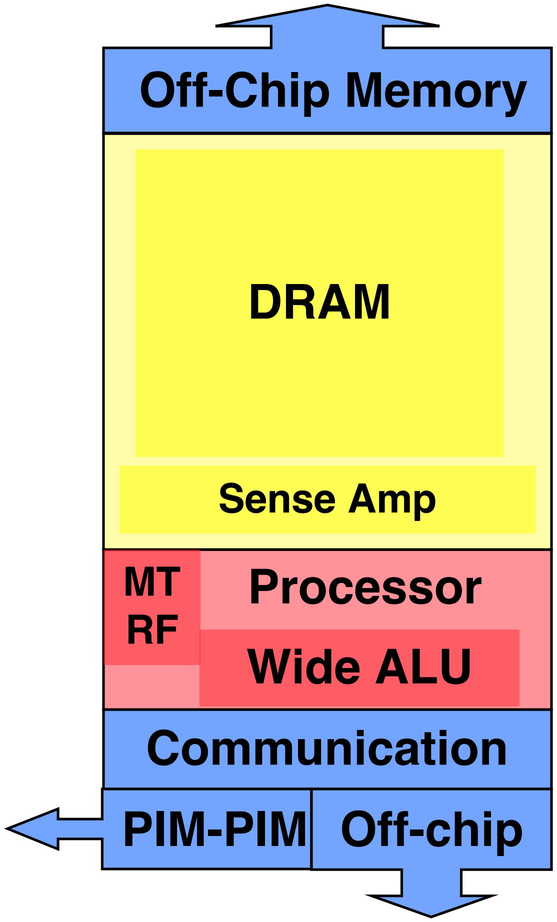






Core-Level View

- Processor + Memory
- Communication
  - On-chip (PIM-PIM)
  - Off-chip (PIM, CPU, More Memory)
- Processor
  - Multithreading?
  - Wide ALU
  - Sense Amp alignment
- Many fabrication options…

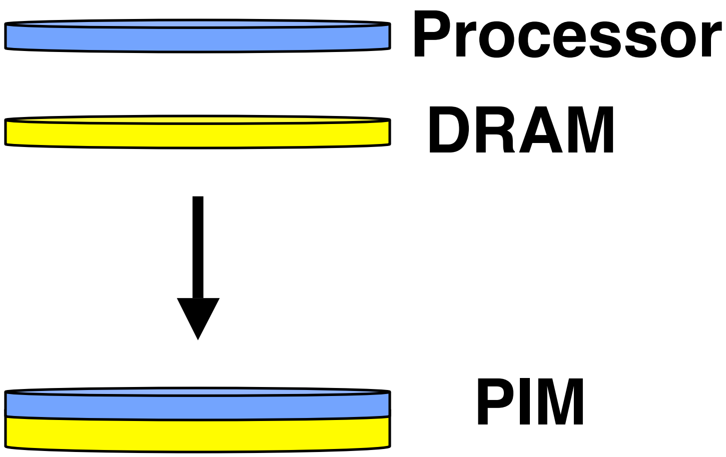


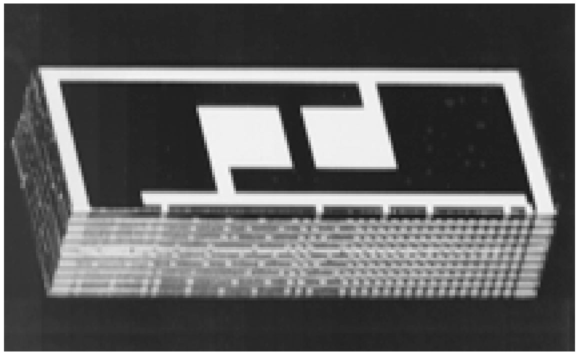
Combined Fabrication

- Processor Fab Process + DRAM Fab process
  - Logic-in-DRAM: Start with DRAM add metal layers
    - Slower DRAM (~5ns), higher density (~1.7 Gb/mm^2)
  - DRAM-in-Logic: Start with logic process, add eDRAM
    - Fast DRAM (~3ns), but DRAM less dense (2.5:1)
- Complexity
  - Add extra steps to fab process -> lower yields?
  - DRAM knowledge & processor knowledge different
  - Design process different

Stacked Fabrication

- DRAM & Logic components fabbed separately
- Dies aligned, joined
- Potential “best of both worlds”
- Uncertainties
  - Alignment process
  - Heat dissipation
  - Die-to-die latency?



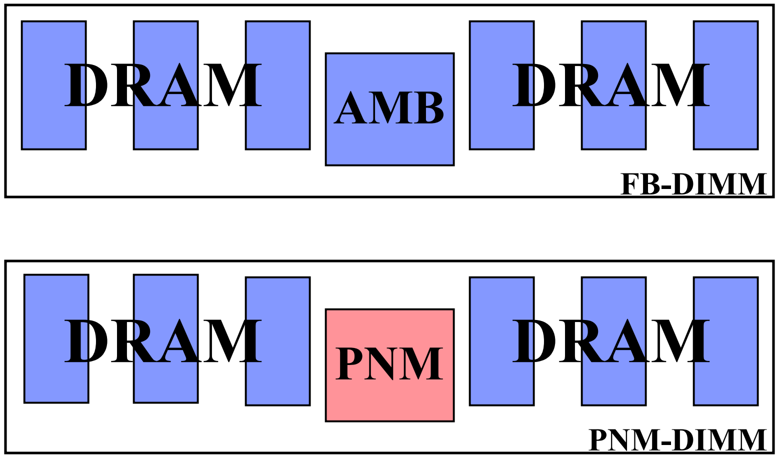


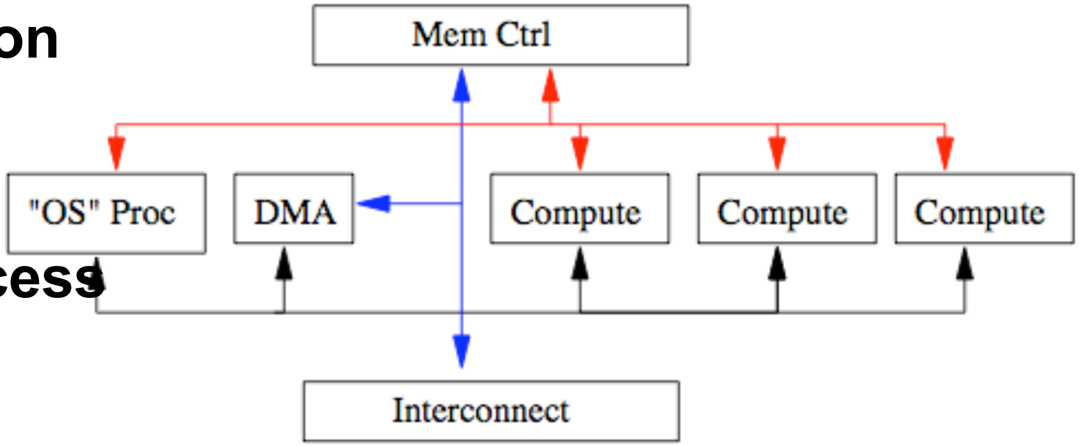
LLNL DRAM “Cube”

Sandia National Laboratories

Processor Near Memory (PNM)

- PNM replaces AMB chip in conventional FB-DIMM
- Multiple compute cores, separate “OS/NIC” processor
  - Low Latency / High Bandwidth
  - Multithreading
  - Hardware Synchronization
- Fabrication Simplicity
  - DRAM in DRAM process
  - Processors in ASIC process
- High efficiency
  - More Flops/mm^2

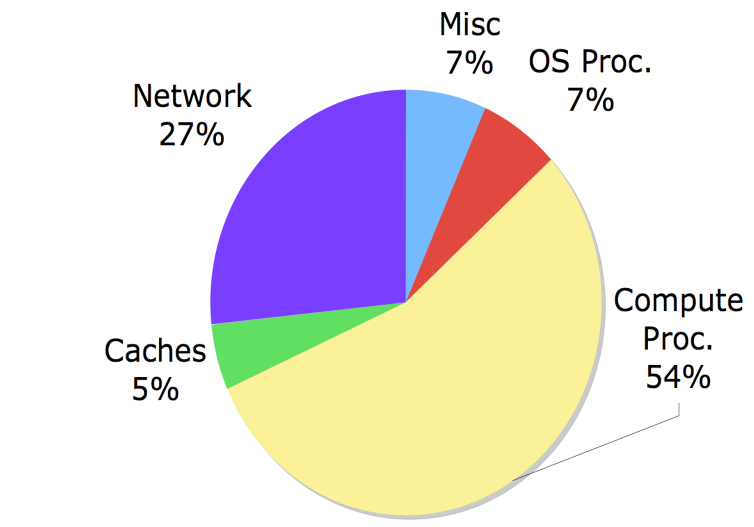




PNM Sizing

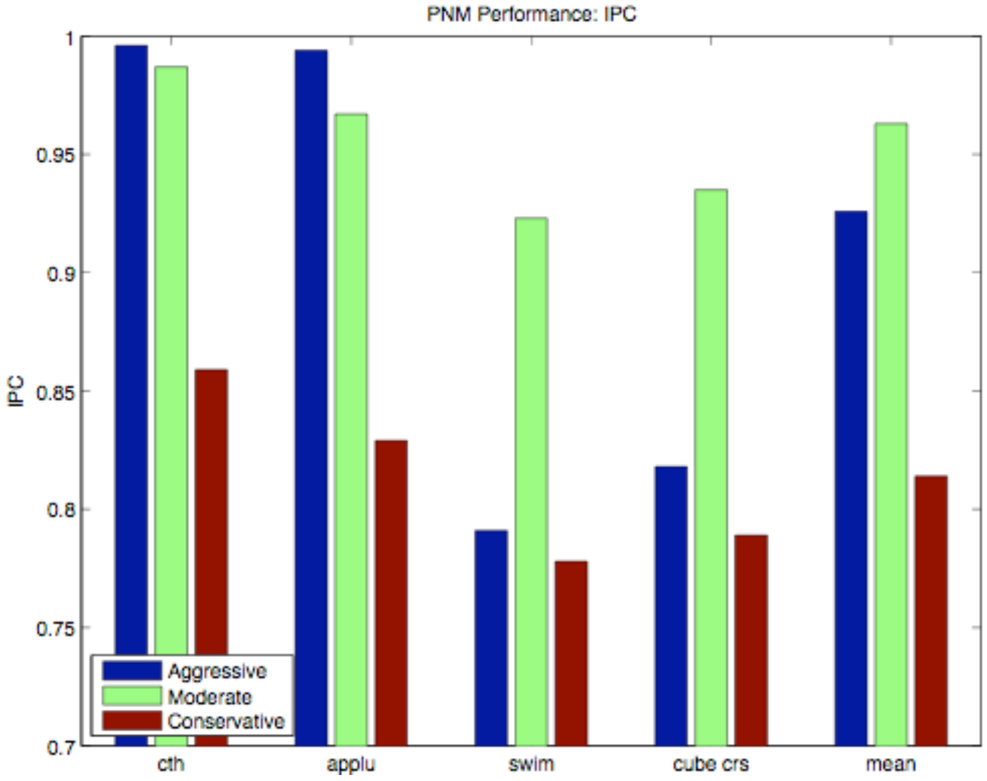
- Goal: Simplicity & Efficiency
- Standard IP Cores
  - DMA, SerDes, Mem. Ctrl
- Network
  - Router-based switching (Dally)
- Caches
  - CACTI
- Processors
  - Based on common embedded cores (ARM, MIPS, etc…)
  - Additional area for MT support: 4.3% * log_2(threads) + RF

	Big	Medium	Little
Off-chip Gbps	20	10	5
Compute Proc.	3 @ 1650 Mhz	2 @ 800 Mhz	2 @ 625 Mhz
Threads	8 / Core	7 / Core	2 / Core
Area mm^2	54 (7.5^2)	22 (4.7^2)	16 ($4^{2)}$



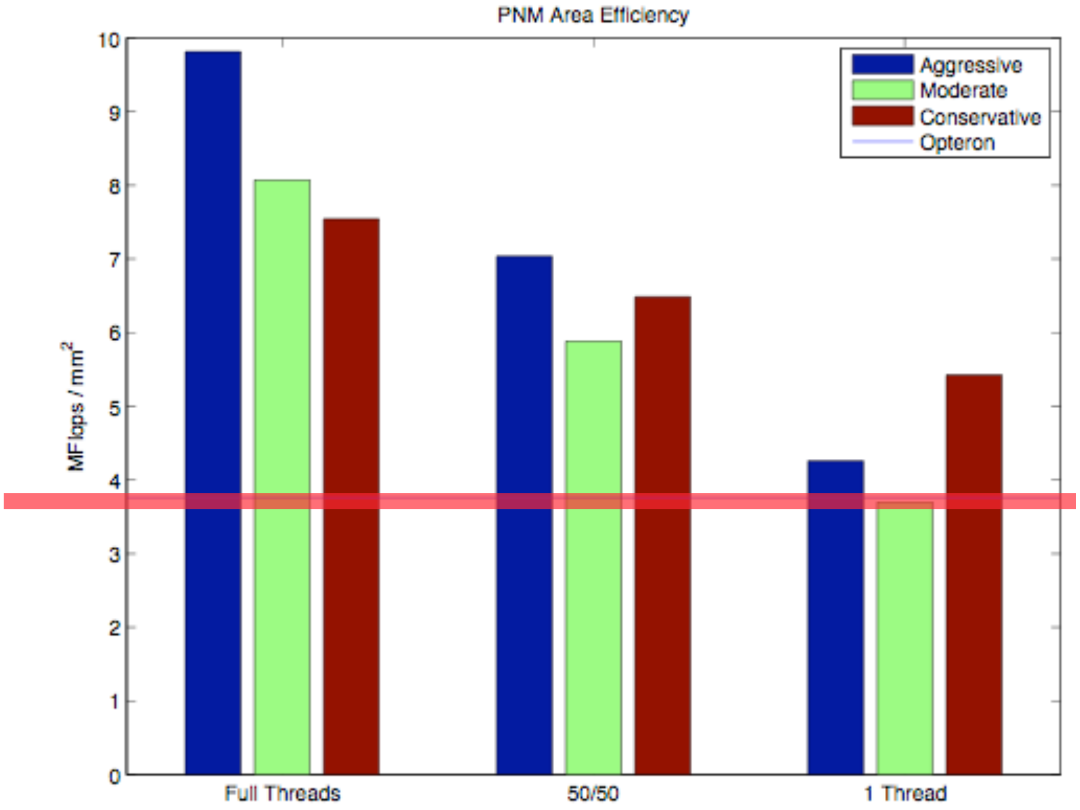
PNM Performance

- Performance simulation performed on Sandia & SPEC apps
- Assume simple (single issue) MT cores
- Low latency to memory (~10 ns)
- Threads cover additional pipeline & memory latency
- Achieve High IPC (>0.75)



PNM Efficiency

- Comparison against 2Ghz Opteron
- Threads critical to performance
  - Saturated threads: 2.0-2.5x more flops/mm^2
  - 50% threads: 1.6-1.9x
  - 1 thread: 1.0-1.4x
- Simple cores, close to memory can beat much more complex cores

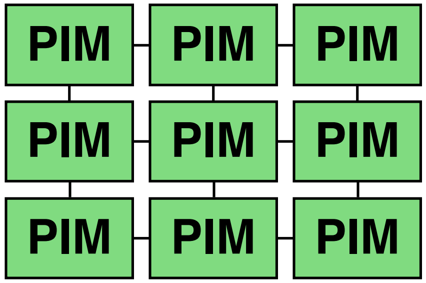


Sandia National Laboratories

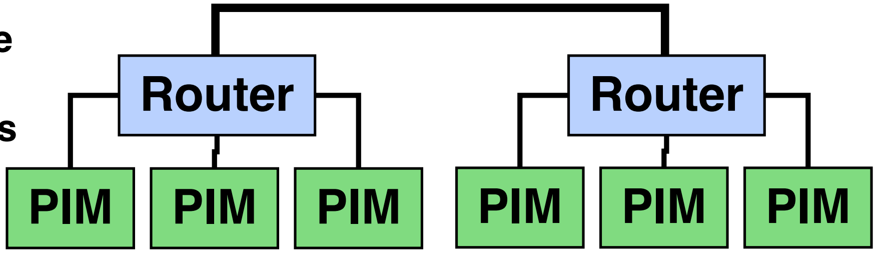
PIM Systems

- “Sea of PIMs”
  - Single Part, elegant
  - Network uncertainty
- “Tree of PIMs”
  - Hierarchical interconnection
- Heterogeneous
  - Conventional CPU + PIMs in MPP configuration
  - Lower risk
- Pure PIM vs. DRAM-backed PIMs

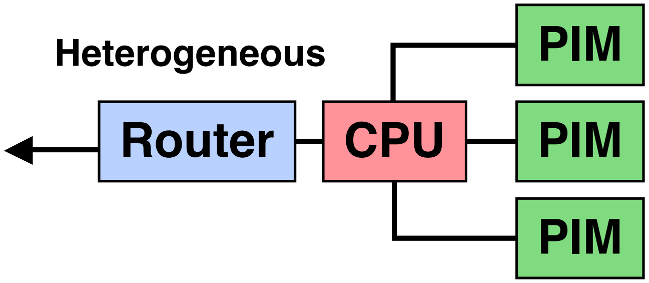
Sea Of PIMs



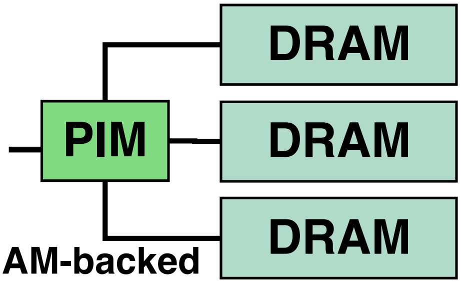
Tree Of PIMs



Heterogeneous



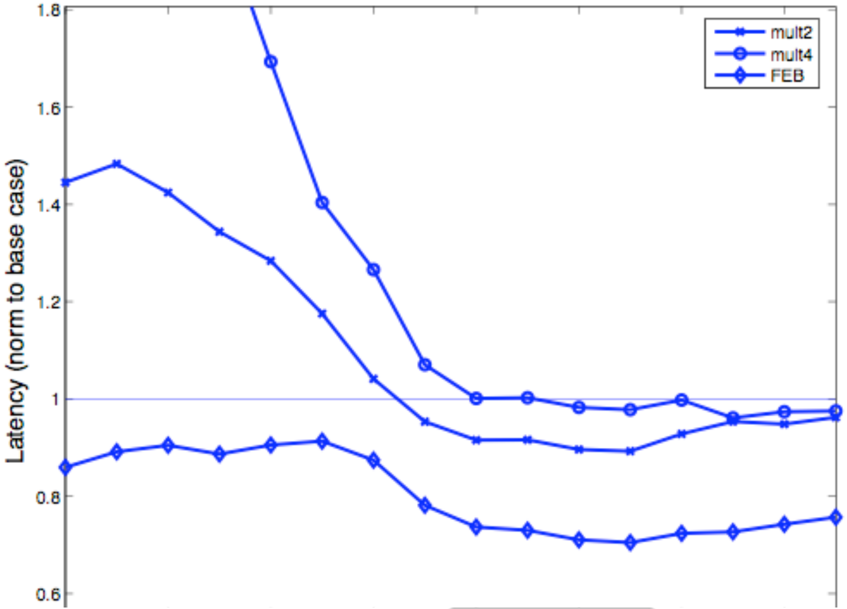
Pure PIM

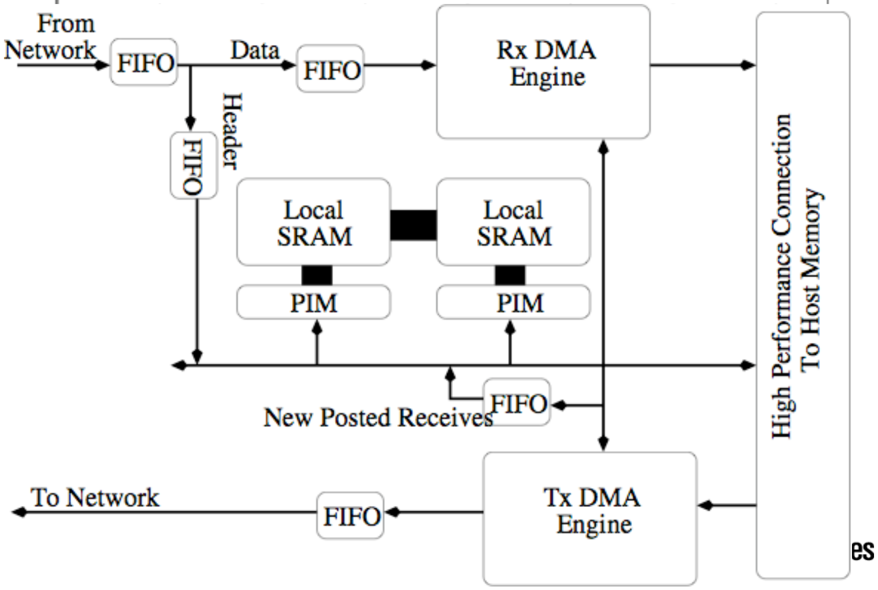


DRAM-backed

Programming: MPI Again?

- Well accepted, understood (we like to think), legacy backing
- PIM offers advantages
  - Wide word, low latency improves message matching
  - Low-level synchronization allow message pipelining
- But…
  - MPI overhead >> shmem
  - Not good at fine-grain parallelism
  - May not be enough





Scatter Gather

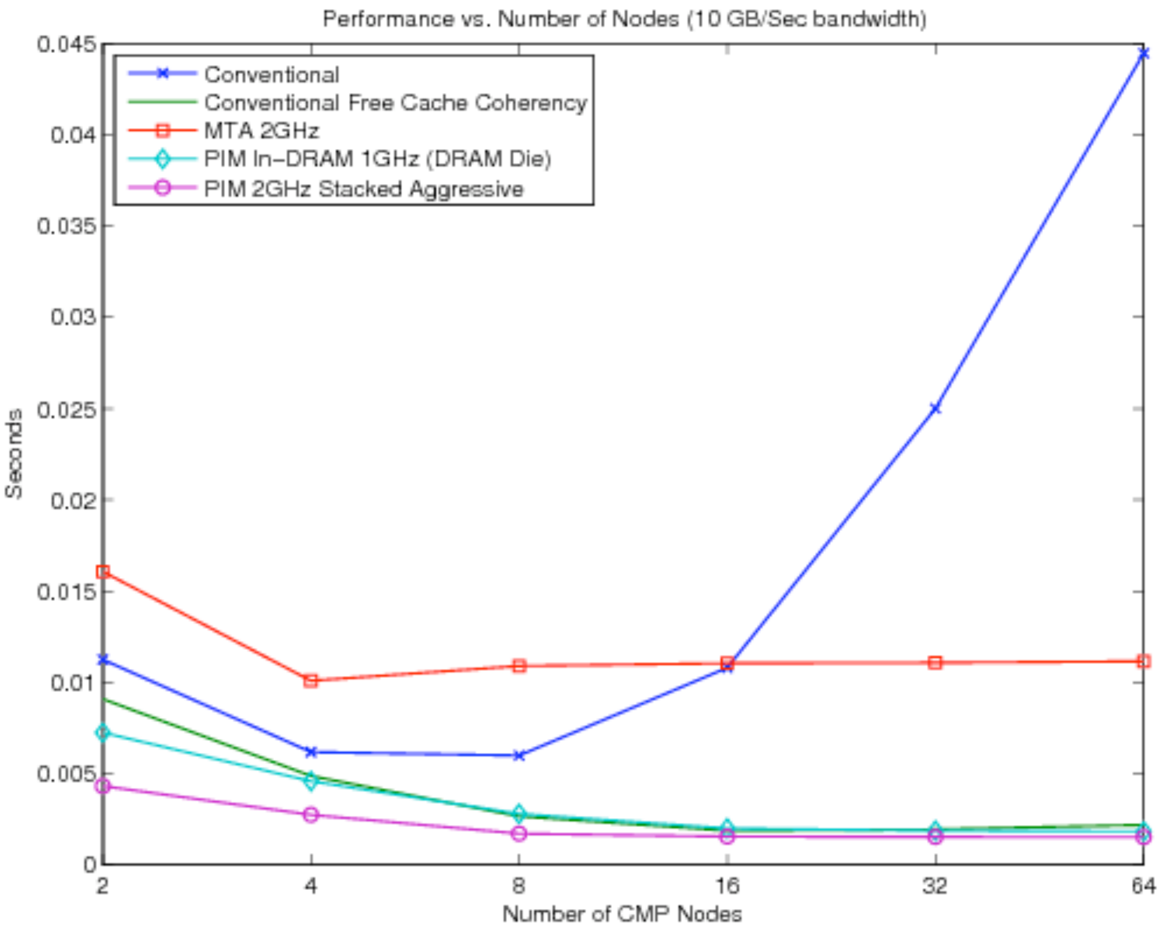
- Integer, memory ops dominate
  - FP ops (“Real work”) < 10% of Sandia codes
  - Several Integer calculations, loads for each FP load
  - Several FP loads per FP op.
- Cray-like Scatter/Gather operations
- Pack data into cache lines, use BW better
- Automatic pointer chasing
  - Graph / list traversal
- Smart prefetching
  - Data collection threads
  - Introspection
- Theme: processing is now cheap, data movement is expensive

Offload/Accelerator

- Augment conventional processor with PIM
- “Hide” PIM programming complexity in library, run-time
- Explicit offload of large “chunks” of computation
- “Master/Slave” model

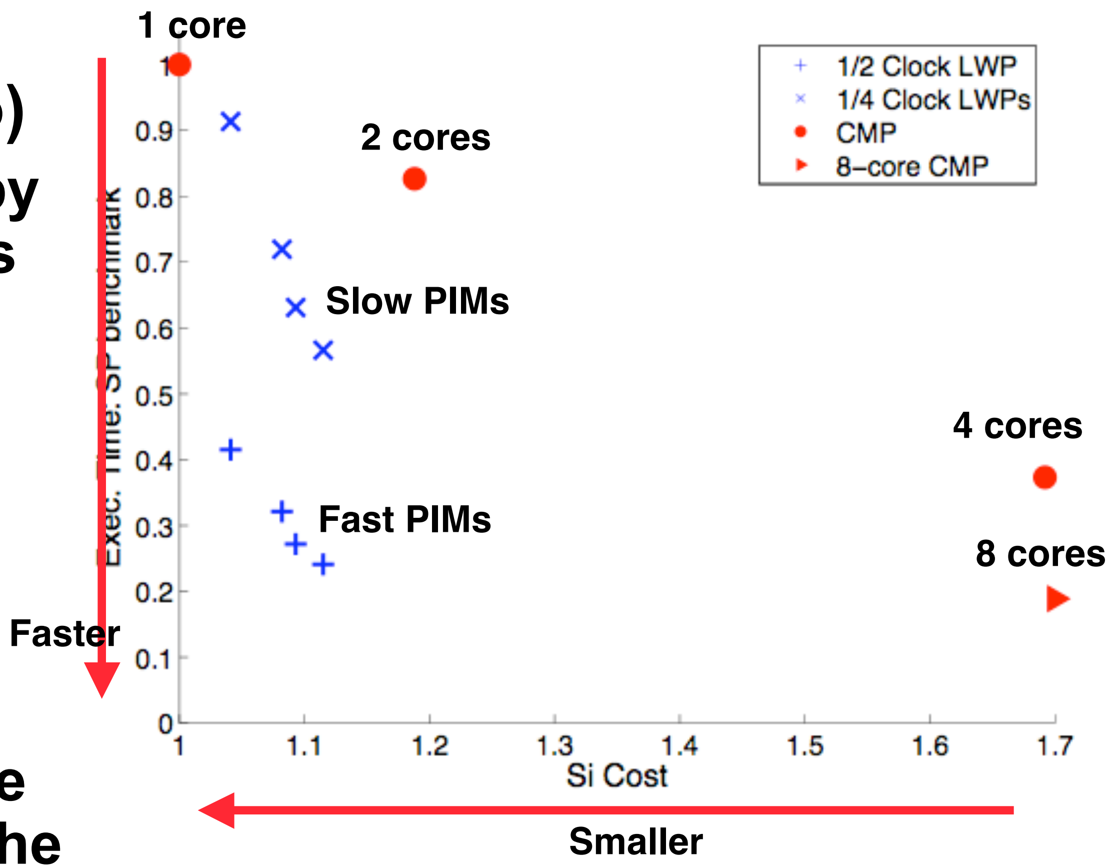
Massive Multithreading

- Multiple levels of parallelism
- Proximity to memory allows fast synchronization
- Highly applicable to certain problems (e.g. Graph)
- Superior scaling & performance for low cost



Multiple Thread Models

- Pursue Parallelism at multiple levels
- Loop-level (OMP, auto)
  - Traditionally limited by expensive processors
  - What if procs were cheap?
- Threadlets
  - Threads w/o stacks, fewer registers
- Migrating threads
  - Move the thread to the data, not the data to the thread

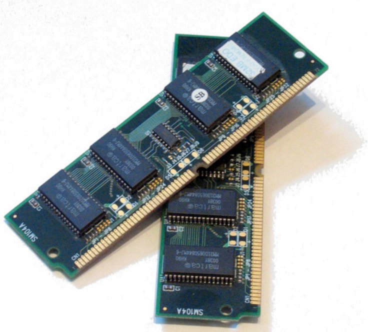


Conclusions

- Cost of processing dropping, cost of data movement still high
- Traditional memory hierarchy complex, filled with bottlenecks – (complex workarounds not working)
- Simplify!
- Can get performance with simple hardware, but need LOTS of parallelism
- Multiple programming models may provide parallelism
- Hardware needs programming model support
- Programming models need hardware support

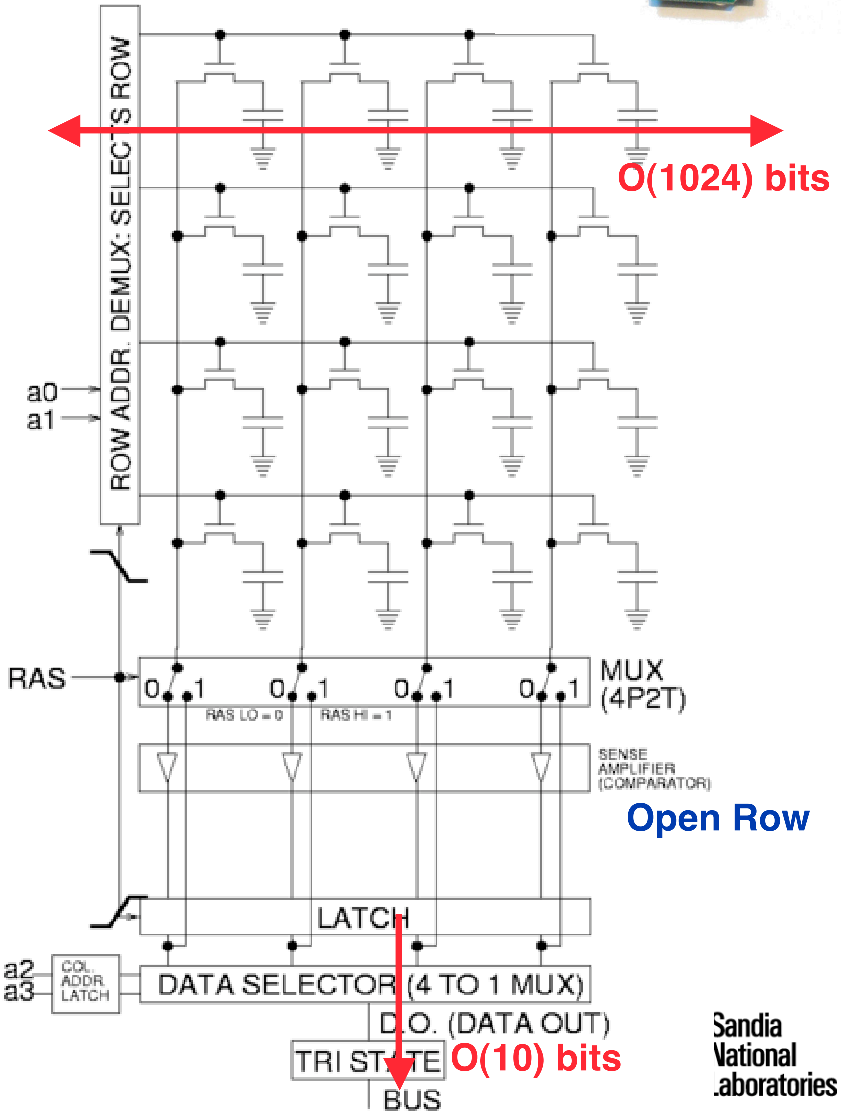
Questions?

Sandia National Laboratories



What a DRAM DIMM Does

- **Matrix of capacitors**
- **Commands**
  - **RAS/CAS: Row/Col Select**
  - **Select, Write, Auto-refresh**
- **Leakage -> Refresh**
- **Latencies**
  - **tRAS: Activate-to-precharge**
  - **tRCD: Row-to-column**
  - **tCAS: Access a column**
  - **tRP: Precharge time**
- **Row reuse key to performance**



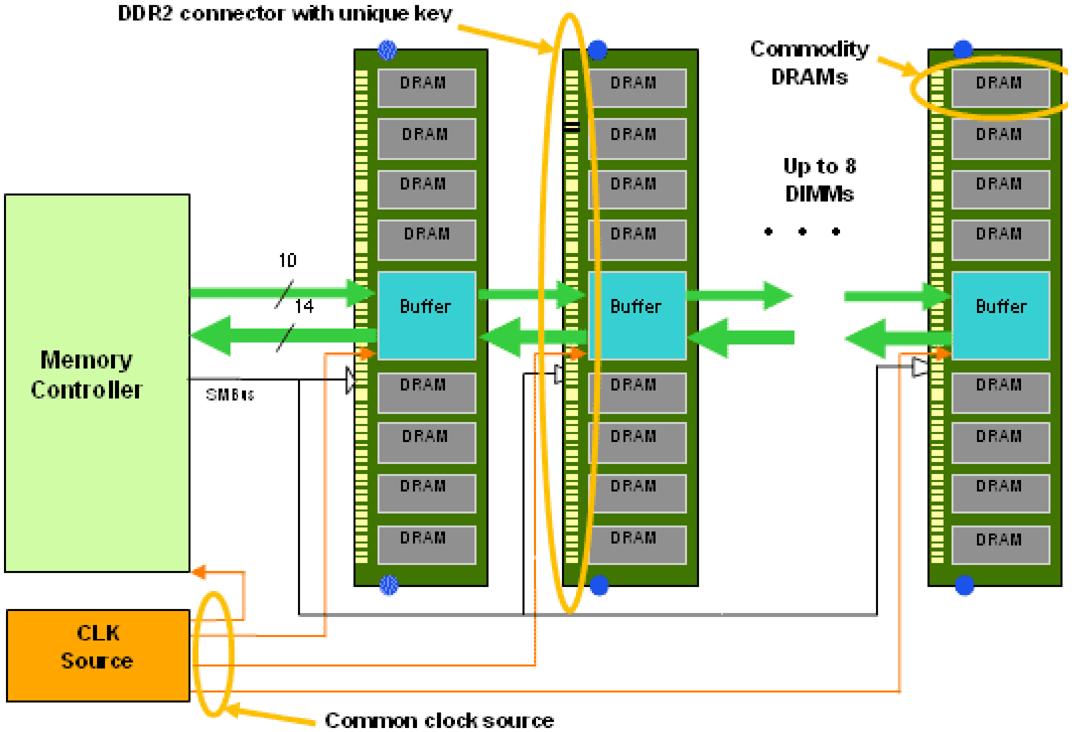
O(1024) bits

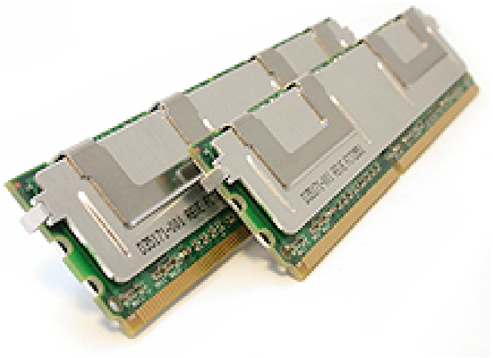
Open Row

O(10) bits

FBDIMM: The Future?

- Goal: increase speed, reliability of DRAM
- Point-to-Point Ring (not bus)
- AMB ASIC controls DIMM
  - Provide error correction
- Faster serial connections
  - More channels
  - Lower pin count
- But…
  - More memory = more latency
  - Cost?





Sandia National Laboratories